CPP 程式設計題

命題者：FGX

題目名稱(中文/英文)：Observation Diary

主要測試觀念：Class Design, Operator Overloading

Basics	Functions
☐ C++ BASICS	☐ SEPARATE COMPILATION AND NAMESPACES
☐ FLOW OF CONTROL	☐ STREAMS AND FILE I/O
☐ FUNCTION BASICS	☐ RECURSION
☐ PARAMETERS AND OVERLOADING	☐ INHERITANCE
☐ ARRAYS	☐ POLYMORPHISM AND VIRTUAL FUNCTIONS
■ STRUCTURES AND CLASSES	☐ TEMPLATES
■ CONSTRUCTORS AND OTHER TOOLS	☐ LINKED DATA STRUCTURES
■ OPERATOR OVERLOADING, FRIENDS, AND REFERENCES	☐ EXCEPTION HANDLING
☐ STRINGS	☐ STANDARD TEMPLATE LIBRARY
☐ POINTERS AND DYNAMIC ARRAYS	☐ PATTERNS AND UML

題目說明：

You are experimenting on several unknown creatures to observe their evolution.

Design class “Creature” to store the status of appendages (body parts) for each creature.

Design class “Diary” to store individual logs kept for each creature.

You may modify everything except the provided “Main.cpp”.

輸入說明：

The main function will be replaced for each test case.

There are several available instructions, while the first line must be NewDay():

```
Diary::NewDay("Date???"); // Change the day of diary
Creature creatureA("CreatureA"); // A new creature named "CreatureA"
Creature creatureB("CreatureB", creatureA); // A clone (body parts) of creatureA named "CreatureB"
creatureA["BodyPartA"] = integerX; // Set the number of "BodyPartA" of "CreatureA"
creatureA["BodyPartA"] += integerY; // Add the number of "BodyPartA" of "CreatureA"
creatureA["BodyPartA"] -= integerZ; // Subtract the number of "BodyPartA" of "CreatureA"
creatureA.PrintStatus(); // Print the current status of creatureA
creatureA.PrintLog(); // Print the log of creatureA
```

輸出說明：

1. Format of PrintStatus(): Look at sample 1.
   Print the name and number of existing appendages (number > 0) sorted by name (string) in ascending order.
2. Format of PrintLog(): Look at sample 2.
   A diary (or log) starts from the target’s creation and is not copied during cloning.
   Log day information when a creature is created and when NewDay() is called.
   Log the change and values when the number of any appendage changes.
   (appear (from zero) / disappear (to zero) / increase (from non-zero) / decrease (to non-zero))

Output a new line after PrintStatus() and PrintLog().

IO 範例：

main	Standard Output

`Diary::NewDay("-4500m");`
`Creature dog("Dog");`
`dog["tail"] = 1;`
`dog["leg"] += 4;`
`dog["antenna"] = 0;`
`dog["head"] = 3;`
`dog.PrintStatus();`
	Dog's status:
	head * 3
	leg * 4
	tail * 1

`Diary::NewDay("00");`
`Diary::NewDay("01");`
`Creature fox("Fox");`
`fox["tail"] += 1;`
`fox["tail"] -= -8;`
`fox["tail"] = 9;`
`Diary::NewDay("10");`
`fox["tail"] += -8;`
`fox["tail"] = 0;`
`Diary::NewDay("11");`
`fox.PrintLog();`
	Fox's log:
	Day 01
	Fox's tail appeared (0 -> 1).
	Fox's tail increased (1 -> 9).
	Day 10
	Fox's tail decreased (9 -> 1).
	Fox's tail disappeared (1 -> 0).
	Day 11

`Diary::NewDay("0000");`
`Creature unknownA("UA");`
`unknownA["leg"] = 16;`
 	
`Diary::NewDay("0102");`
`Creature unknownB("UB", unknownA);`
`unknownB["leg"] += 26;`
`unknownA.PrintLog();`
 	
`Diary::NewDay("0227");`
`unknownA["leg"] = 0;`
`unknownA.PrintStatus();`
`unknownB.PrintLog();`
 	
`Diary::NewDay("0353");`
`unknownA["leg"] += 6;`
`unknownA["wing"] += 4;`
`unknownA.PrintLog();`
	UA's log:
	Day 0000
	UA's leg appeared (0 -> 16).
	Day 0102
	
	UA's status:
	
	UB's log:
	Day 0102
	UB's leg increased (16 -> 42).
	Day 0227
	
	UA's log:
	Day 0000
	UA's leg appeared (0 -> 16).
	Day 0102
	Day 0227
	UA's leg disappeared (16 -> 0).
	Day 0353
	UA's leg appeared (0 -> 6).
	UA's wing appeared (0 -> 4).

附屬資料：

☑解答程式：Creature.cpp, Creature.h, Diary.cpp, Diary.h

☑測試資料：Main1.cpp, output1.txt, Main2.cpp, output2.txt, Main3.cpp, output3.txt

■易，僅需用到基礎程式設計語法與結構

□中，需用到多項程式設計語法與結構

□難，需用到多項程式結構或較為複雜之資料型態或結構

解題時間：17 分鐘

其他註記：

Semantics for sample 2: Fox's dairy started from "Day 01" and ended on "Day 11".

Trust that you, the observer, never make any status error (e.g. negative number of appendages).